APLIKASI INFORMATION RETRIEVAL UNTUK PEMBENTUKAN TESAURUS BERBAHASA INDONESIA SECARA OTOMATIS

Cholifah[1)], Yudhi Purwananto[2)], Arif Bramantoro[2)]

[1,2,3)] *Teknik Informatika, FTIF, ITS, Surabaya email: c_ifa@yahoo.com*

Abstract, *In Information Retrieval, a thesaurus which could provide a term list with its similar terms can be used to search a document within a collection of documents. With the growth of information, a thesaurus is expected to help more in finding information so that more relevant document could be retrieved. The purpose of this research is to find a method to form an automatic thesaurus generation. This process requires a term dictionary in specific field and a group of documents to perform calculation defining the relationship within its existing terms. The generation of the thesaurus is done by calculating the paired-occurrence value within the terms which is found in a collection of documents. Theexperiment is done by using some respondents to define the terms that are relevant with a specific term. The result showed that the system could provide accuracy in generating the thesaurus with the average recall value of 59.62 % and the average precision of 66.78 %.*

Keywords: *Information Retrieval, automatic thesaurus generation, similarity, recall, Indonesian thesaurus*

Pertumbuhan yang sangat pesat pada internet dan *World Wide Web* menyebabkan pula berkembang pesatnya sumber daya informasi elektronik. Hal ini telah memunculkan masalah baru yaitu bagaimana mencari dan mendapatkan informasi yang dibutuhkan. Oleh karena itu suatu cabang ilmu baru telah muncul dalam Teknologi Informasi yaitu *Information Retrieval* (Temu Kembali Informasi).

Tesaurus adalah bentuk yang berharga dalam sistem pencarian informasi. Sebuah tesaurus akan menyediakan daftar kata yang tepat dan terkontrol yang berguna dalam mengkoordinasikan pengindeksan maupun pencarian dokumen. Tesaurus telah digunakan dalam dunia informasi untuk memecahkan masalah ketidakkonsistenan pada pengindeksan dokumen, dan juga dapat digunakan oleh pencari dalam memformulasi ulang strategi pencarian yang tepat jika diperlukan.

Pembangunan tesaurus berbahasa Indonesia disini, didasari oleh kebutuhan akan pencarian informasi berbahasa Indonesia. Pembangunan tesaurus dapat dilakukan dengan dua cara yaitu dengan cara manual atau dengan cara otomatis.

Pembangunan Tesaurus secara otomatis dilakukan dengan melakukan analisa terhadap kemunculan pasangan kata(*co-word*) dalam kumpulan dokumen. Oleh karena itu diperlukan pembelajaran (*learning*) untuk mengelompokkan kata berbahasa Indonesia dengan tingkat kemiripan tertentu berdasar frekuensi kemunculan pasangan kata dalam koleksi dokumen.

Kemampuan tesaurus dalam menemukan kata akan sangat tergantung dengan ruang lingkup pembahasan, dimana **eksekusi** pada ruang lingkup hukum akan sangat berbeda maknanya dibanding dengan **eksekusi** pada ruang lingkup teknologi komputer. Oleh karena itu, pada penelitian ini penulis menggunakan sebuah ruang lingkup pembahasan yaitu bidang teknologi informasi dan komputer. Kumpulan dokumen yang digunakan dalam pembangunan tesaurus diambil dari berbagai sumber seperti makalah ilmiah, berita, artikel, paper, dan sebagainya.

Tujuan dari penelitian ini adalah membangun perangkat lunak yang mampu menentukan tesaurus dari kata berbahasa Indonesia di bidang teknologi informasi dan komputer yang dientrikan

SISTEM TEMU KEMBALI INFORMASI

Pengertian

Salah satu pengertian dari IR menurut *Bill Frakes* dan *Ricardo Baeza-Yates* adalah "sub

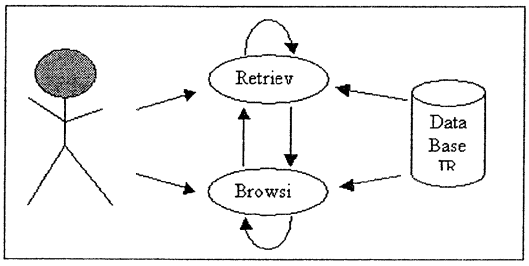
bidang dari ilmu komputer yang mempelajari tentang pengumpulan data dan temu kembali dokumen". Dalam perkembangan selanjutya IR dikembangkan menjadi *automated IR systems* yang menangani temu kembali data secara otomatis dalam jumlah data yang besar.

Konsep Dasar

Keefektifan dari temu kembali informasi yang diinginkan tergantung pada dua hal mendasar yaitu perilaku pengguna dan *logical view* dari sistem temu kembali.

Perilaku Pengguna

Dalam sistem IR, pengguna menterjemahkan kebutuhan informasinya ke dalam bentuk kata kunci (*keyword*), lalu *keyword* ini oleh sistem IR akan diterjemahkan menjadi suatu *query*, dengan *query* ini maka sistem akan melakukan pencarian ke dalam kumpulan dokumen. Kualitas informasi yang dihasilkan dari sistem IR, secara langsung dipengaruhi dua hal yaitu penyajian dokumen secara logikal oleh sistem dan perilaku pengguna, perilaku pengguna maksudnya adalah berhubungan dengan kegiatan pengguna menentukan kata kunci yang sesuai dengan informasi yang diinginkan. Interaksi antara pengguna dengan sistem IR melalui aktifitas yang berbeda dapat digambarkan seperti dibawah ini.



Gambar 1. Interaksi antara pengguna dengan sistem IR melalui aktifitas yang berbeda

View Dokumen

Sistem basisdata saat ini memungkinkan menyimpan representasi dokumen dalam bentuk koleksi keseluruhan kata-kata yang dikandungnya. Dalam hal ini dikatakan bahwa sistem IR mengadopsi *full text logical view.* Dengan banyaknya dokumen yang harus diproses, maka akan semakin besar kapasitas database yang diperlukan. *Full text* merupakan penampakan paling lengkap dari suatu dokumen tetapi penggunaannya membutuhkan biaya komputasi yang tinggi.

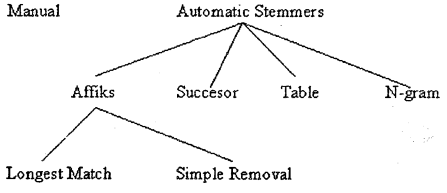
Untuk mengatasi hal tersebut, sistem harus dapat mereduksi kata-kata yang disimpan dan mentransformasi logical view dari full text menjadi bentuk indeks. Hal ini dilakukan dengan meng-eliminasi stoplist (kata-kata yang terlalu umum seperti kata sandang, kata sambung, kata ganti, dll) dan melakukan proses stemming (pengubahan bentuk inbuhan ke bentuk kata dasar). Lebih jauh lagi adalah penggunaan metode-metode kompresi teks.

Operasi-operasi dalam Information Retrieval

Operasi pada *Information Retrieval* dapat dibagi menjadi 3 bagian besar yaitu :
Operasi *query*
Operasi kata(term)
Operasi dokumen.

Metode Stemming

Salah satu teknik untuk meningkatkan performa sistem IR adalah pencarian berdasarkan variasi morfologi kata. Salah satu contohnya untuk kata 'siap' akan lebih mudah dikenali dalam suatu dokumen yang besar dibanding kata 'mempersiapkan'. *Stemming* digunakan untuk mencari kata dasar dari bentuk berimbuhan, misalkan untuk kata lari, pelari, berlari, dilarikan, melarikan, semuanya akan dihitung menjadi satu *term* dengan frekuensi lima, bukan menjadi lima *term* berbeda dengan frekuensi masing-masing satu. Secara umum metode *stemming* yang sampai saat ini telah dikembangkan dapat dilihat pada gambar berikut.



Gambar 2. Pembagian Metode Stemming

Dalam penelitian ini digunakan metode *stemming* Affiks Removal Stemmers, yang dilakukan dengan menghilangkan awalan dan atau akhiran dari kata-kata berimbuhan.

Kelemahan metode ini adalah tidak bisa menghilangkan sisipan dari suatu kata berimbuhan, seperti kata 'pemerintah' dari kata dasarnya 'perintah'. Kata ini akan tetap dikenali sebagai 'pemerintah'. Algoritma yang dikembangkan oleh Porter sangat bagus untuk mendapatkan hasil maksimal dalam proses *stemming*, namun algoritma ini memerlukan waktu yang sangat panjang karena operasi yang dilakukan adalah per karakter dalam satu kata sebelum didapatkan hasil yang diinginkan. Secara garis besar algoritma yang digunakan untuk metode *stemming* dalam Penelitian ini akan dijelaskan sebagai berikut ; Contoh pemenggalan kata "mempersiapkannya" :

- Langkah 1 :
  Cek apakah kata ada dalam kamus
  Ya : Success
  Tidak : lakukan pemotongan awalan (mem)
  Kata = permainkannya
- Langkah 2 :
  Cek apakah kata ada dalam kamus
  Ya : Success
  Tidak : lakukan pemotongan akhiran (nya)
  Kata = permainkan
- Langkah 3 :
  Cek apakah kata ada dalam kamus
  Ya : Success
  Tidak : lakukan pemotongan awalan (per)
  Kata = mainkan
- Langkah 4 :
  Cek apakah kata ada dalam kamus
  Ya : Success
  Tidak : lakukan pemotongan akhiran (kan)
  Kata = main
- Langkah 5 :
  Cek apakah kata ada dalam kamus
  Ya : Success
  Tidak : Kembalikan kata ke bentuk asli.
  Increment frekuensi kata.

Penggunaan algoritma ini cukup efektif dalam pengubahan bentuk berimbuhan menjadi kata dasar (*stemming)*, namun masih adanya fenomena *overstemming* yang mengakibatkan pemotongan bagian kata yang sebenarnya adalah milik kata dasar itu sendiri yang kebetulan mirip dengan salah satu jenis imbuhan yang ada (dalam jumlah kecil).

PEMBANGUNAN TESAURUS

Umum

Dalam *information retrieval*, tesaurus dapat digunakan untuk membantu pengindeksan dan penemuan kebali informasi karena tesaurus dapat menyediakan kosakata yang tepat dan terkontrol sehingga pengguna dapat memilih istilah yang tepat untuk kedua penggunaan diatas. Dalam pencarian, pengguna dapat menggunakan tesaurus untuk mendesain strategi pencarian yang paling tepat.

Jika pencarian tidak menghasilkan dokumen yang cukup, tesaurus dapat digunakan untuk memperluas *query* dengan mengikuti *link* yang terdapat diantara term. Dengan cara yang sama, jika pencarian menghasilkan dokumen yang terlalu banyak, tesaurus dapat memberi alternatif perbendaharaan kata yang lebih spesifik untuk pencarian selanjutnya.

Pembangunan Tesaurus Secara Otomatis

Menurut Frakes dan Yates

Frakes dan Yates (Frakes, 1992) secara garis besar menentukan tiga fase dalam pembangunan tesaurus secara otomatis, yaitu:

1. Penyusunan daftar kata
   - Normalisasi dan pemilihan term
   - Pembangunan frase sesuai dengen level koordinasi yang diinginkan
2. Perhitungan tingkat kemiripan (*similarity*)
   - Mengidentifikasi hubungan antar term secara statistik
3. Pengorganisasian daftar kata
   - Mengorganisasikan kosa kata secara umum menjadi susunan hirarki berdasar hubungan yang telah terkomputasi pada langkah (2).

Menurut Schubert Foo

Proses tesaurus menurut Schubert Foo dapat dibagi menjadi dua sub fase yaitu fase pembentukan kamus dan fase pembangunan tesaurus.

Fase Pembentukan Kamus Kata

Tujuan pembangunan kamus adalah membantu dalam mengekstrak term yang spesifik dengan domain(misalnya term di bidang ilmu komputer) dalam koleksi dokumen untuk pembentukan tesaurus kemudian. Langkah yang dilakukan:

- Term Selection (Pemilihan Term)
- Term Filtering
- Term Specification/Generalisation

Fase Pembentukan Tesaurus

Dengan adanya kamus yang telah dibentuk pada fase sebelumnya, kemudian

digunakan metode yang diusulkan oleh Chen[3] untuk membangun tesaurus akhir.

- **Menghitung Frekuensi Term dan Frekuensi Dokumen.**
  Pada saat ekstraksi, dilakukan dua perhitungan frekuensi untuk setiap term. Frekuensi tersebut adalah frekuensi term, tfij, yaitu jumlah kemunculan term j di dokumen i, dan frekuensi dokumen, dfj, yaitu jumlah dokumen dimana terdapat term j.
- **Menghitung Bobot Term**
  Bobot term j pada dokumen i, dij, dihitung dengan menggunakan rumus:

$$dij = tfij \times \log\left(\frac{N}{dfj} \times lj\right)$$

Dimana :
tfij = jumlah kemunculan term j pada dokumen I
N = jumlah total dokumen pada koleksi dokumen
dfj = jumlah dokumen dimana term j muncul.
lj = jumlah kata pada term(kata atau frase)

- **Analisa Asymmetric Co-occurrence**
  Bobot kemiripan / *similarity weight* (atau *cluster weight*) dari term *Tj* terhadap term *Tk:*

$$ClusterWeight(Tj,Tk) = \frac{\sum_{i=1}^{n} dijk}{\sum_{i=1}^{n} dij} \times WeightingFactor(Tk)$$

dijk merupakan bobot kombinasi dari kedua term *Tj* dan *Tk* pada dokumen *i*, yang didefinisikan dengan rumus berikut:

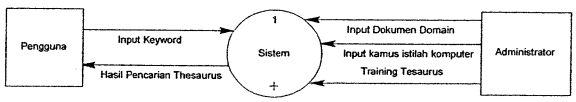
$$dijk = tfijk \times \log\left(\frac{N}{dfjk} \times lj\right)$$

dimana:
tfijk = jumlah kemunculan kedua term *j* dan term *k* dalam dokumen *i*(nilai yang terkecil diantara keduanya yang diambil).
dfjk = jumlah dokumen dimana term *j* dan *k* muncul bersamaan
Sedangkan WeightingFactor digunakan untuk menyaring term yang terlalu umum pada saat analisa *co-ocurence*.
Rumus yang digunakan :

$$WeightingFactor(Tk) = \frac{\log \frac{N}{dfk}}{\log N}$$

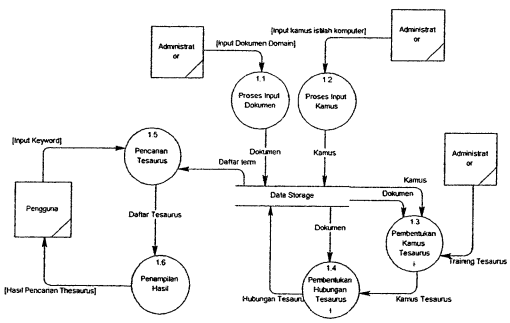
METODOLOGI

Proses serta alur data pada pembentukan tesaurus otomatis dapat dilihat pada data flow diagram berikut :



Gambar 3. Gambar DFD Level 0

proses diatas dapat di *breakdown* menjadi DFD level 1 berikut :



Gambar 4. Gambar DFD Level 1

PROSES PEMBENTUKAN KAMUS

- *Term Selection*
  Proses ini melakukan pemilihan terhadap kamus istilah komputer yang telah dimasukkan sebelumnya. Jika istilah tersebut tidak pernah dipakai sama sekali dalam dokumen, maka istilah tersebut dihapus dari daftar kamus.
- *Term Filtering*

Bahwa proses *term filtering* diperlukan untuk melengkapi kamus tesaurus dengan menambahkan term-term yang tidak terdapat pada kamus sebelumnya.

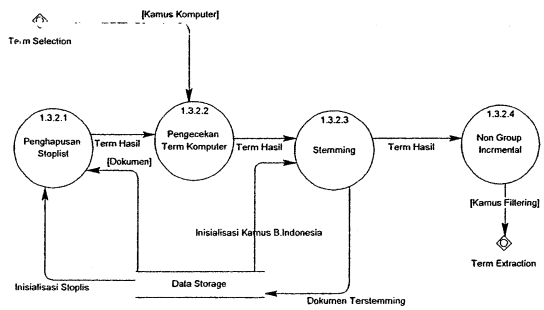
- *Term Extraction*
  Proses ini melakukan penggabungan term yang didapatkan dari proses *term filtering* maupun term selection menjadi kamus yang digunakan sebagai acuan pembentukan tesaurus kemudian.

Sub Proses Term Filtering

Term filtering merupakan langkah alternatif lain pembentukan kamus tesaurus yang bertujuan menambah daftar kamus tesaurus dengan term-term yang tidak terdapat dalam kamus komputer sebelumnya.

Metode yang baik diperlukan agar dapat mengambil term dari dokumen yang bisa dianggap sebagai pembeda. Metode yang kurang bagus dapat menghasilkan jumlah term yang berlimpah sehingga akan semakin memperlambat proses komputasi pada saat perhitungan similarity.

Proses term filtering dijelaskan dengan gambar diagram alir berikut :



Gambar 5. Diagram alir data proses term filtering

Proses Pembentukan Tesaurus

Langkah yang dilakukan :

- **Hitung Frekuensi Term dan Frekuensi Dokumen**
  Proses ini melakukan perhitungan terhadap frekuensi kemunculan masing-masing term dan pasangan term pada masing-masing dokumen
- **Hitung Bobot Term**
  Proses ini melakukan perhitungan bobot term dan pasangan term pada masing-masing dokumen. Penjumlahan bobot masing-masing term dan pasangan term di seluruh dokumen akan diperlukan untuk perhitungan tingkat kemiripan antar term.
- **Analisa Co-occurrence**
  Proses ini akan melakukan perhitungan terhadap kemunculan berpasangan suatu term dalam setiap dokumen yang terdapat dalam koleksi dokumen untuk pembelajaran. Hasil dari proses ini adalah bobot kemiripan pada masing-masing pasangan term.

PEMBAHASAN

Implementasi Data

Data-data yang diperlukan untuk pembentukan aplikasi yang berupa data masukan, maupun data keluaran, semuanya disimpan dalam bentuk dokumen teks (*.txt).

Data lain yang dibutuhkan pada inisialisi awal program adalah file "Kamus.txt", yang berisi daftar istilah komputer dan teknologi informasi. Untuk dapat diterima oleh sistem, daftar istilah ditulis dengan huruf kapital dan tiap istilah dipisahkan dengan karakter [Enter].

Implementasi Proses

Proses pembentukan tesaurus dalam tugas akhir ini dibagi lagi menjadi lima sub proses di bawah ini:

Filter Kamus

Proses filtering dilakukan dengan menghapus istilah yang tidak pernah digunakan sama sekali dalam koleksi dokumen training. Proses *filtering* ini dilakukan untuk mengefisiensikan proses dan waktu komputasi.

Menghitung Bobot Term

Proses ini dilakukan untuk menghitung jumlah total bobot sebuah term pada masing-masing dokumen, di seluruh koleksi dokumen. Hasil akhir dari proses ini adalah file dokumen *dfj.txt* dan *dij.txt*. File *dfj.txt* berisi data yaitu jumlah frekuensi dokumen dimana terdapat suatu term. Jadi file ini menyimpan data frekuensi dokumen dari masing-masing term yang terdapat dalam kamus tesaurus.

Hasil keluaran yang kedua yaitu file *dij.txt* menyimpan hasil perhitungan bobot masing-masing term di tiap koleksi dokumen training. Hasil yang disimpan dalam file tersebut berupa penjumlahan/sigma bobot tiap

term pada masing-masing dokumen di seluruh koleksi dokumen.

Menghitung Weighting Factor

Seperti telah dijelaskan pada bab 2, data *Weighting Factor(Wf)* diperlukan untuk melakukan perhitungan *Cluster Weight*

Menghitung Bobot Pasangan Term

Proses ini melakukan perhitungan bobot semua kombinasi pasangan term pada tiap dokumen, kemudian menjumlahkan bobot masing-masing pasangan term diseluruh koleksi dokumen dan menyimpannya pada file teks *SigmaDij.txt*.

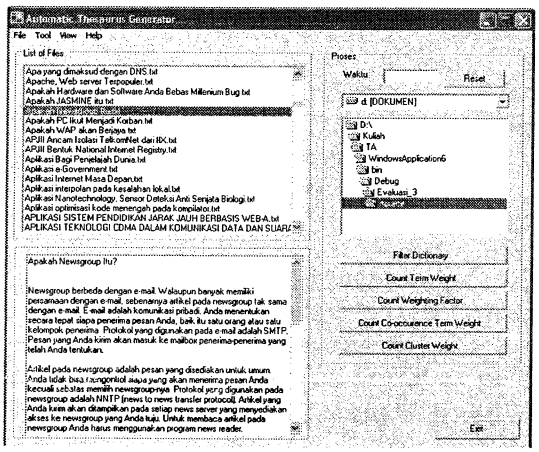
Menghitung Cluster Weight

Cluster Weight adalah hasil akhir dari proses pembelajaran program untuk perhitungan bobot kemiripan antar term yang terdapat pada kamus tesaurus.

Hasil akhir dari proses ini adalah dokumen teks yang menyimpan data *cluster weight* setiap kombinasi pasangan term yang terdapat pada kamus tesaurus. Nilai *cluster weight* setiap pasangan kata disimpan dalam file teks *similarity.txt*.

IMPLEMENTASI ANTARMUKA

Secara garis besar, antar muka program pembentuk hubungan tesaurus otomatis adalah seperti tampak pada gambar berikut :



Gambar 6. Tampilan Antar Muka Aplikasi

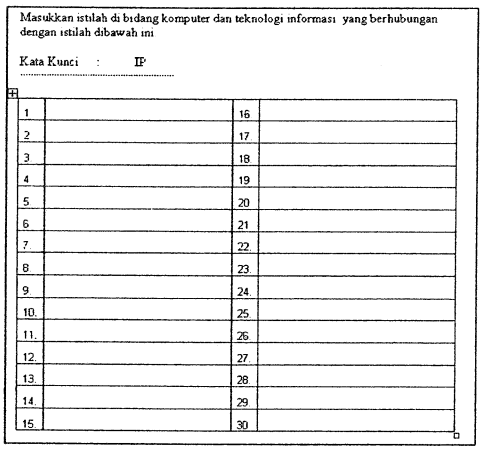
UJI COBA

Dalam pembuatan kamus awal, daftar istilah (term) dikumpulkan dari beberapa kamus komputer yang diambil dari beberapa sumber, kemudian digabung menjadi satu kamus yang lebih lengkap dengan jumlah total term sebanyak 2426 term. Pada tahap selanjutnya, dilakukan penghapusan terhadap term yang tidak pernah digunakan dalam dokumen sehingga term berhasil disortir menjadi 1102 term.

Koleksi dokumen yang digunakan untuk proses pembelajaran diambil dari abstrak makalah teknik, berita dan artikel online. Jumlah total dokumen yang digunakan adalah 1196 dokumen dengan ± 200 dokumen berasal dari makalah teknik dan sisanya adalah artikel dan berita tentang teknologi komputer.

Dari data tersebut, proses pembelajaran sistem menghasilkan 1.102 term dan 109.151 pasangan term (pasangan dengan bobot *cluster weight*<>0). Untuk kebutuhan interaksi pengguna dan sistem yang produktif, sebuah uji coba diperlukan untuk menentukan threshold yang membatasi jumlah maksimal term yang ditampilkan pada setiap query agar menghasilkan term dengan hasil relevansi yang terbaik. Uji coba dilakukan dengan mengambil 10 term yang terdapat dalam kamus tesaurus secara acak. Masing-masing term ini kemudian digunakan sebagai masukan pada sistem untuk dicari tesaurusnya. Dari 10 term contoh ini kemudian ditentukan term-term yang dianggap relevan dengan masing-masing term contoh tersebut. Untuk itu digunakan 10 orang respondenuntuk menentukan term-term yang dianggap relevan.

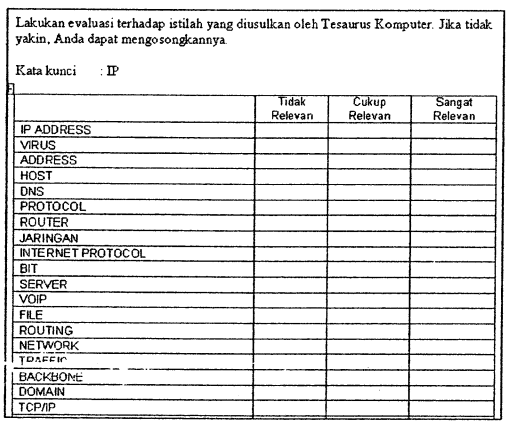
Langkah pertama dalam menentukan term yang relevan adalah dengan memberikan masing-masing responden sebuah kuisioner dengan sebuah istilah komputer, kemudian responden diminta untuk memasukkan istilah-istilah yang dianggap berhubungan dengan istilah yang telah dipilih. Selanjutnya, responden diberikan lembar kuisioner kedua untuk memberikan penilaian terhadap istilah yang dihasilkan oleh sistem tesaurus dengan menentukan istilah-istilah yang dianggap tidak relevan, cukup relevan, dan sangat relevan. Bentuk kuisioner adalah tampak seperti pada gambar 7 dan 8.


Masukkan istilah di bidang komputer dan teknologi informasi yang berhubungan dengan istilah dibawah ini

Kata Kunci : IP

1		16	
2		17	
3		18	
4		19	
5		20	
6		21	
7		22	
8		23	
9		24	
10		25	
11		26	
12		27	
13		28	
14		29	
15		30	

Gambar 7. Kuisioner pertama yang digunakan dalam uji coba penentuan relevant term


Lakukan evaluasi terhadap istilah yang diusulkan oleh Tesaurus Komputer. Jika tidak yakin, Anda dapat mengosongkannya.

Kata kunci : IP

	Tidak Relevan	Cukup Relevan	Sangat Relevan
IP ADDRESS			
VIRUS			
ADDRESS			
HOST			
DNS			
PROTOCOL			
ROUTER			
JARINGAN			
INTERNET PROTOCOL			
BIT			
SERVER			
VOIP			
FILE			
ROUTING			
NETWORK			
TRAFFIC			
BACKBONE			
DOMAIN			
TCP/IP			

Gambar 8. Kuisioner kedua yang digunakan dalam uji coba penentuan relevant term

Evaluasi dilakukan dengan melakukan perhitungan *concept recall* dan *concept precision*. [2]. Perhitungan dilakukan dengan menggunakan rumus sebagai berikut :

$$\text{Concept Recall (r)} = \frac{\text{Number of Retrieved Relevant Concept}}{\text{Number of Total relevant Concept}}$$

$$\text{Concept Precision (p)} = \frac{\text{Number of Retrieved Relevant Concept}}{\text{Number of Total Retrieved Concept}}$$

Gambar 9. Rumus perhitungan Concept Recall dan Precision

Total Relevant Concept adalah term yang dihasilkan pada kuisioner 1 ditambah dengan term pada kuisioner dua yang bernilai "sangat relevan" dan "cukup relevan".

Setelah perhitungan *Recall* dan *Precision*, untuk evaluasi digunakan juga perhitungan rata-rata harmonik(F1) dengan rumus :

$$F_1 = \frac{2}{\frac{1}{r} + \frac{1}{p}}$$

Gambar 10. Rumus perhitungan Rata-rata Harmonik

Rumus ini digunakan untuk mencari nilai kompromi dari *recall* dan *precision*, dimana nilai F1 yang maksimal dapat dianggap sebagai kemungkinan kompromi yang terbaik diantara *recall* dan *precision*, sehingga threshold yang diambil untuk tesaurus adalah yang mempunyai nilai F1 yang tertinggi.

Uji coba dilakukan dengan mengambil 10 term secara acak untuk dihitung nilai recall dan precision. Term yang digunakan untuk uji coba adalah Algoritma Genetika, Assembly, Backbone, Bahasa Pemrograman, Active Desktop, Basis Data, BIOS, Celeron, Citra, Driver. Lima macam kemungkinan nilai *cluster weight* telah dipilih sebagai threshold yaitu 0.08, 0.1, 0.15, 0.2, 0.25, dimana threshold yang menghasilkan nilai F1 yang tertinggi yang akan digunakan dalam penentuan hubungan tesaurus selanjutnya.

Dari data yang telah didapatkan pada uji coba sebelumnya, kemudian dihitung nilai F1 rata-rata untuk setiap threshold pada semua term sampel. Nilai F1 rata-rata dapat digambarkan pada table

Tabel 1. Nilai rata-rata Recall, Precision dan F1

Threshold	Average Recall	Average Precision	Average F1
0.08	0.667807844	0.625655781	0.572580128
0.1	0.596166175	0.667770028	0.574832897
0.15	0.426631852	0.735584102	0.493851784
0.2	0.318326483	0.769475146	0.406917053
0.25	0.245881842	0.775297619	0.358533391

Dari tabel hasil uji coba diatas, diketahui bahwa hasil pembentukan tesaurus paling baik didapatkan saat menggunakan threshold *cluster weight* = 0.1(F1=0.5748), yang menghasilkan nilai Avarage Recall 0.5962 (59.62 %), dan Average Precision sebesar 0.6678 (66.78 %). Hasil ini kemudian akan dapat digunakan sebagai acuan dalam pembentukan hubungan

tesaurus pada istilah dibidang komputer dan teknologi informasi.

SIMPULAN

1. Pembentukan tesaurus secara otomatis dapat dilakukan dengan menggunakan metode pengelompokan term menggunakan asymmetric cluster function, yang mampu menghasilkan hubungan tesaurus diantara istilah dengan tingkat keberhasilan Avarage Recall 59.62 % dan Avarage Precision 66.78 % .
2. Penggunaan kamus di bidang komputer dan teknologi informasi secara keseluruhan di fase awal pembentukan tesaurus sangat berguna dalam menghilangkan intervensi manual dalam pembentukan tesaurus. Hal ini dapat mungkin dilakukan dengan adanya langkah seleksi kamus (term selection) yang menghapus secara otomatis istilah-istilah dalam kamus komputer yang tidak pernah digunakan dalam dokumen.
3. Proses yang seragam dan tidak membedakan istilah yang bermakna sama, sehingga perhitungan frekuensi istilah dan dokumen yang dilakukan terpisah dapat menyebabkan ketidakakuratan dalam perhitungan bobot kemiripan (misalnya pada term *prosesor* dan *processor, atau pada term harddisk* dan *hard disk*).

DAFTAR RUJUKAN

[1] Eric Butow and Tommy Ryan, "*C#*", Hungry Minds, Inc, New York, 2002.

[2] Gerard Salton, "*Automatic Text Processing : The Transformation, Analysis, and Retrieval of Information by Computer*", Addison-Wesley, 1989.

[3] Hsinchun Chen, Bruce Schatz, Joanne Martinez, Tobun Dorbin Ng. *"Generating a Domain-Specific Thesaurus Automatically:An Experiment on FlyBase"*. Information Processing and Management, 1997. *http://ai.bpa.arizona.edu/papers/fly94/fly94.html*

[4] Hsinchun Chen and Kevin J. Lynch. *"Automatic Construction of Network of Concepts Characterizing Document Database"*. MIS Department, University of Arizona Tucson, Arizona. 1994.

[5] *http://citeseer.nj.nec.com/chen94automatic.html*

[6] Jack Febrian dan Farida Andayani, *"Kamus Komputer dan Istilah Teknologi Informasi"*, Penerbit Informatika, Bandung, 2002.

[7] Scubert Foo, Siu Cheung Hui, Hong Koon Lim, and Li Hui. *"Automatic Thesaurus for Enhanched Chinese Text Retrieval"*. School of Applied Science, Nanyang Technological University, Nanyang Avenue, Singapore, 2000. *http://islab.sas.ntu.edu.sg:8000/user/schubert/publications/2000/00libreview_fmt.PDF*

[8] Ricardo Baeza-Yates, Berthier Ribeiro-Neto, *"Modern Information Retrieva"l*, ACM Press, New York, Addison-Wesley Longman Limited, 1999.

[9] William B. Frakes and R. Baeza-Yates, *"Information Retrieval : Data Structures and Algorithms"*, Prentice Hall, 1992